Fuzzy Pooling

Aditya Suryawanshi
DA-675 Fuzzy Systems & Applications ,
210150004

***Abstract*—Convolutional Neural Networks (CNNs) are artificial learning systems typically based on two operations: convolution, which implements feature extraction through filtering, and pooling, which implements dimensionality reduction. The impact of pooling in the classification performance of the CNNs has been highlighted in several previous works, and a variety of alternative pooling operators have been proposed. However, only a few of them tackle with the uncertainty that is naturally propagated from the input layer to the feature maps of the hidden layers through convolutions. The paper presents a novel pooling operation based on (type-1) fuzzy sets to cope with the local imprecision of the feature maps, and we investigate its performance in the context of image classification. Fuzzy pooling is performed by fuzzification, aggregation and defuzzification of feature map neighborhoods. It is used for the construction of a fuzzy pooling layer that can be applied as a drop-in replacement of the current, crisp, pooling layers of CNN architectures. Several experiments using publicly available datasets show that the proposed approach can enhance the classification performance of a CNN. A comparative evaluation shows that it outperforms state-of-the-art pooling approaches.**

***Index Terms*—Convolutional Neural Networks, Image analysis, Classification, Pooling, Fuzzy sets.**

I. Introduction

Convolutional Neural Networks (CNNs) have significantly advanced the field of computer vision, enabling remarkable performance in tasks such as image classification, object detection, and segmentation. A crucial component of CNNs is the pooling layer, which reduces the spatial dimensions of feature maps, thus minimizing computational load. Traditional pooling methods, such as max pooling and average pooling, have been widely adopted; however, they often overlook the uncertainty and imprecision inherent in visual data. To address these limitations, fuzzy pooling methods have been proposed, which incorporate fuzzy logic principles to enhance the pooling operation. Fuzzy logic, introduced by Professor Zadeh in 1965, allows for reasoning under uncertainty and has been widely used in various applications, including control systems, decision-making, and data analysis. The incorporation of fuzzy logic into neural networks offers a way to model the uncertainty in data more effectively than traditional methods. This is particularly relevant in domains where data can be imprecise or vague, such as image processing.

In Convolutional Neural Networks (CNNs), pixel values of input images are susceptible to various uncertainties, such as noise and ambiguities in color and geometry. As this uncertain information is propagated through the network layers, it affects the feature maps generated by each layer. Since convolution operates locally, the uncertainty in the feature map's values can also be regarded as local

II. Literature Review

Fuzzy Pooling Techniques Fuzzy pooling techniques extend the conventional pooling mechanisms by utilizing fuzzy sets to represent the uncertainty in feature maps. Various methods have been proposed in the literature, each with unique approaches to integrate fuzzy logic into the pooling layer.

Fuzzy Max Pooling: One of the earliest approaches, fuzzy max pooling, replaces the hard decision-making of max pooling with a fuzzy logic framework. Instead of selecting a single maximum value, fuzzy max pooling considers a set of potential maxima, weighted by their membership functions in fuzzy sets. This method can retain more informative features and improve the model's robustness to noise and distortion .

Fuzzy Average Pooling: Fuzzy average pooling combines the outputs of the fuzzy max pooling with the average pooling technique, allowing for a more comprehensive representation of the feature maps. By utilizing membership functions to compute weighted averages, this method helps to mitigate the loss of important spatial information, leading to improved feature extraction .

Dynamic Fuzzy Pooling: Dynamic fuzzy pooling introduces an adaptive mechanism that adjusts the pooling parameters based on the input data characteristics. This approach allows the pooling operation to be more context-aware, enhancing the network's ability to learn from diverse datasets. Research has shown that dynamic fuzzy pooling can lead to improved performance in image classification tasks, particularly in scenarios with significant variations in input data .

Fuzzy Spatial Pooling: Another innovative approach is fuzzy spatial pooling, which considers the spatial relationships between pixels when performing pooling operations. By leveraging spatial information, fuzzy spatial pooling can better preserve the structural integrity of images, leading to enhanced feature representation and ultimately improving classification accuracy .

Fuzzy pooling techniques offer several advantages over traditional pooling methods:

Handling Uncertainty: Fuzzy pooling effectively models uncertainty and imprecision in visual data, leading to more robust feature extraction. Improved Information Retention: By utilizing membership functions, fuzzy pooling can retain more critical information from feature maps, enhancing the performance of CNNs. Adaptability: Techniques such as dynamic fuzzy pooling allow models to adaptively learn from different data distributions, improving generalization capabilities.

III. Methodology

To model the uncertainty of the input layer , fuzzy sets can be employed, represented as

$$\tilde{y}_v = \{\langle x, \mu_v(x)\rangle \mid x \in E\}, \quad v = 1, \ldots, V \tag{1}$$

where these sets capture overlapping value intervals that can be linguistically characterized (e.g., small, medium, and large). Here, the universe E is determined based on the output value ranges of the neural activation functions from the convolutional layer.

A. Choice of Activation Function

ReLU is defined as

$$\mathrm{ReLU}(x) = \max(0, x) \quad (x \in \mathbb{R}) \tag{2}$$

It is usually preferred over conventional sigmoid activations because it is computationally simpler and reduces the possibility of vanishing gradients, which deep neural networks often suffer from.

For this paper capped ReLU is used and is defined as:

$$\mathrm{ReLU}(x, r_{\max}) = \min(\max(x, 0), r_{\max}) \tag{3}$$

This formulation introduces a maximum value $r_{\max}$ to the ReLU function, enabling it to maintain output within a specified range while still leveraging the advantages of the ReLU activation.

B. Patch Size for Pooling

Let β represent a volume of size $w \times h \times z$, consisting of z feature maps β_n each of size $w \times h$, i.e., $\beta = \{\beta_n \mid n = 1, 2, \ldots, z\}$.

The first step of the proposed methodology involves sampling the input volume using a pooling window of size $k \times k$. Common choices for the hyperparameters are $k = 3$ and $\sigma = 2$, which typically reduce the width and height of the input volume by half. This process yields a set of volume patches from the input volume β with a stride of σ. Each volume patch consists of spatial patches p_n extracted from the feature maps β_n, represented as $p = \{p \mid n = 1, 2, \ldots, z\}$.

The total number of patches c that can be extracted from the input volume β is calculated as:

$$c = \frac{(w - k + 2t_w)(h - k + 2t_h)}{2\sigma + 2} \tag{4}$$

where $t_w = \frac{(\sigma-1)(w-\sigma+k)}{2}$ and $t_h = \frac{(\sigma-1)(h-\sigma+k)}{2}$ are the zero-padding values used in the patch extraction process along the width and height axes, respectively.

Let p_n denote an element of a volume patch p at depth n and position (i, j), where $i = 1, \ldots, k$, $j = 1, \ldots, k$, and $n = 1, \ldots, z$.

For simplicity, we consider three fuzzy sets defined by (1) for $V = 3$. These fuzzy sets $\tilde{y}_1, \tilde{y}_2$, and $\tilde{y}_3$ correspond to small, medium, and large values of p_n, with membership functions μ_1, μ_2, and μ_3, respectively.

C. Membership Functions

The first membership function is defined as:

$$\mu_1(p^n_{i,j}) = \begin{cases} 0 & \text{if } p^n_{i,j} > d \\ \frac{d - p^n_{i,j}}{d - c} & \text{if } c \le p^n_{i,j} \le d \\ 1 & \text{if } p^n_{i,j} < c \end{cases} \tag{5}$$

where $d = \frac{r_{\max}}{2}$ and $c = \frac{d}{3}$.

The second membership function is defined as:

$$\mu_2(p^n_{i,j}) = \begin{cases} 0 & \text{if } p^n_{i,j} \le a \\ \frac{p^n_{i,j} - a}{m - a} & \text{if } a \le p^n_{i,j} \le m \\ \frac{b - p^n_{i,j}}{b - m} & \text{if } m < p^n_{i,j} < b \\ 0 & \text{if } p^n_{i,j} \ge b \end{cases} \tag{6}$$

where $a = \frac{r_{\max}}{4}$, $m = \frac{r_{\max}}{2}$, and $b = m + a$.

The third membership function is defined as:

$$\mu_3(p^n_{i,j}) = \begin{cases} 0 & \text{if } p^n_{i,j} < r \\ \frac{p^n_{i,j} - r}{q - r} & \text{if } r \le p^n_{i,j} \le q \\ 1 & \text{if } p^n_{i,j} > q \end{cases} \tag{7}$$

D. Fuzzy Patches[Fuzzification]

For each patch p_n, where $n = 1, \ldots, z$, a fuzzy patch π^n_v is defined as:

$$\pi^n_v = \mu_v(p^n) = \begin{pmatrix} \mu_v(p^n_{1,1}) & \cdots & \mu_v(p^n_{1,k}) \\ \vdots & \ddots & \vdots \\ \mu_v(p^n_{k,1}) & \cdots & \mu_v(p^n_{k,k}) \end{pmatrix} \tag{8}$$

E. Score formulation for each fuzzy patch[Aggregation]

Pooling begins with the spatial aggregation of the values of the fuzzy patch, using the fuzzy algebraic sum operator $\dot{\Sigma}$, as follows:

$$s^n_{\pi_v} = \sum_{i=1}^{k} \sum_{j=1}^{k} \pi^n_{v,i,j}, \quad n = 1, \ldots, z. \tag{9}$$

This operator was selected as a standard s-norm, considering all the neighboring values of the fuzzy patch. It has a relatively low computational complexity and can be easily vectorized for efficient execution on a GPU. The value of each s_n is considered as a score quantifying the overall membership of p to $\tilde{y}_v$.

F. Final Fuzzy volume Patch

Based on these scores, for each volume patch p, a new fuzzy volume patch π' is created by selecting the spatial fuzzy patches π_n for $v = 1, \ldots, V$ that have the largest scores s_n, i.e.,

$$\pi' = \{\pi'_n = \pi_n \mid v = \arg\max(s_n), n = 1, 2, \ldots, z\} \tag{10}$$

G. Output of Fuzzy pooling[Defuzzification]

This way, patches of higher certainty are selected. The dimensionality of each patch is then reduced by defuzzification using the Center of Gravity (CoG):

$$p'_n = \frac{\sum_{i=0}^{k}\sum_{j=0}^{k}\left(\pi'_{n,i,j}\cdot p_{n,i,j}\right)}{\sum_{i=0}^{k}\sum_{j=0}^{k}\pi'_{n,i,j}} \quad (11)$$

where $p' = \{p'_n \mid n = 1, 2, \ldots, z\}$.

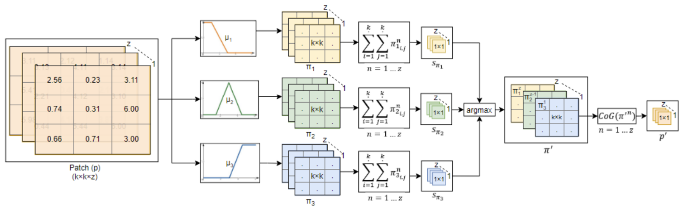

Fig. 1: Illustration of fuzzy pooling operation in a CNN.

IV. Results and Discussions

A. About the Dataset

The classification performance of the proposed method was evaluated using the MNIST dataset, which contains 70,000 grayscale images of handwritten digits (0-9), sized 28×28 pixels. Comprising 60,000 training and 10,000 test images, the dataset serves as a standard benchmark for assessing classification algorithms.

B. Classification Model description

To minimize hyperparameter bias, we evaluated the proposed pooling method using the LeNet baseline CNN architecture. LeNet, with its relatively simple structure and fewer parameters compared to modern deep learning models, allows for clearer isolation of the pooling method's impact on performance. Training was conducted using Stochastic Gradient Descent (SGD) with a batch size of 32 images. Additionally, no data preprocessing or augmentation was applied, ensuring that the evaluation focused solely on the effect of the pooling layer on classification outcomes.

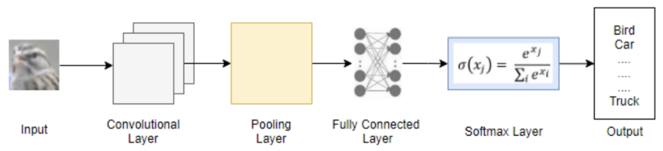

Fig. 2: Illustration of Simple LeNet architecture used.

In this study, the impact of the fuzzy pooling layer on model accuracy was measured on the MNIST test set, with results indicating improved performance compared to Average pooling and Max Pooling as shown in Table 1.This test was performed using stride 2 for taking patches . With better system configurations and larger data size to train on the results for fuzzy pooling further improve .

Method	Classification Accuracy (%)
Max Pooling	98.69
Average Pooling	98.10
Proposed Fuzzy Pooling	98.80

TABLE I: Comparison of Classification Accuracy for Different Pooling Methods on MNIST dataset

V. Conclusions and Future Scopes

This study introduced a novel fuzzy pooling method for CNNs designed to handle the inherent uncertainty in feature values. The proposed fuzzy pooling approach significantly improved classification performance, outperforming traditional pooling methods by enhancing the network's ability to manage data variability and uncertainty. Experiments on standard image datasets validated that fuzzy pooling retains important features in pooled areas more effectively than existing methods, leading to higher classification accuracy and stronger generalization.

For future work, the fuzzy pooling method could be extended and optimized for deeper, more complex CNN architectures, exploring its potential across diverse tasks beyond image classification, such as object detection and segmentation. Additionally, integrating fuzzy pooling with other uncertainty-handling techniques, like Bayesian CNNs, could further enhance model robustness in high-variance environments. Investigating its adaptability to different data types, including audio and video, and optimizing its computational efficiency will also be valuable to maximize the impact and applicability of fuzzy pooling in real-world applications.

References

[1] Dimitrios E. Diamantis, and Dimitris K. Iakovidis, Senior Member, IEEE, , Fuzzy Pooling, 2020.
[2] Naila Murray and Florent Perronnin , Generalised Max Pooling, CVPR, 2014.
[3] Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine, 29(6), 141–142.
[4] Yann LeCun et al , LeNet Architecture for CNN, 1989